

We claim:

1        1. Apparatus for forming an identifier for an input  
2        object and for securely marking the input object with the  
3        identifier so as to yield a marked object, the apparatus  
4        comprising:

5              a processor; and  
6              a memory having computer executable instructions  
7        stored therein; and

8              wherein the processor, in response to the stored  
9        executable instructions:

10             generates a flow representation for the input  
11        object, the representation having a plurality of nodes,  
12        said nodes representing predefined first operations  
13        performed by the input object, and connections among the  
14        nodes signifying associated flow among the predefined  
15        first operations performed by the input object;

16             randomly selects first and second nodes from  
17        the plurality of nodes in the representation so as to  
18        form a pre-defined number of nodal pairs, each of said  
19        pairs having one of the first nodes and a corresponding  
20        one of the second nodes; and

21             for each of the nodal pairs, establishes flow  
22        between the first and second nodes in said each nodal  
23        pair and inserts, in the flow so established, a selected  
24        one of a plurality of different pre-defined second  
25        operations so as to collectively define the marked  
26        object, whereby the marked object implements the  
27        predefined first operations and a plurality of selected  
28        ones of the predefined second operations, each of which  
29        has been randomly spliced into flow of the input object,

30 wherein the identifier collectively comprises all the  
31 different ones of the plurality of predefined second  
32 operations, and the associated execution flow associated  
33 therewith and involving the nodal pairs.

1 2. The apparatus in claim 1 wherein the input object  
2 comprises a software object.

1 3. The apparatus in claim 2 wherein the software object  
2 comprises, input executable code, at least one  
3 instruction in the input executable code is associated  
4 with a corresponding one of the predefined first  
5 operations, and executable code for a corresponding  
6 executable procedure is associated with each selected one  
7 of the predefined second operations.

1 4. The apparatus in claim 3 wherein the processor, in  
2 response to the stored instructions:

3 inserts a pre-defined number of separate links and  
4 designations for the selected ones of the procedures into  
5 the flow representation so as to yield a combined flow  
6 representation; and

7 converts, in response to said input executable code  
8 and executable code for said selected ones of the  
9 procedures, said combined flow representation into output  
10 executable code, said output executable code being the  
11 marked code.

1 5. The apparatus in claim 4 wherein the input  
2 executable code comprises first and second portions  
3 thereof and the flow representation comprises first and

4 second separate flow representations for the first and  
5 second portions of the input executable code,  
6 respectively.

1 6. The apparatus in claim 5 wherein:

2 the first portion of the input executable code  
3 comprises at least a pre-defined portion of a non-marked  
4 application program; and

5 the second portion of the input executable code  
6 comprises a remaining portion of the non-marked  
7 application program or pre-defined executable security  
8 code.

1 7. The apparatus in claim 6 wherein the processor, in  
2 response to the stored instructions, inserts executable  
3 code for the selected one procedure in noncontiguous  
4 locations in the input executable code.

1 8. The apparatus in claim 6 wherein the processor, in  
2 response to the stored instructions, selects the  
3 procedure from a pre-defined library of stored routines,  
4 wherein said procedure is one of the stored routines.

1 9. The apparatus in claim 8 wherein each of the  
2 inserted procedures implements, when executed, a  
3 pre-defined function such that if any of said inserted  
4 procedures is removed from the marked code, the marked  
5 code, when subsequently executed, will terminate its  
6 execution.

DRAFTED: FEB 22 1990

1       10. The apparatus in claim 8 wherein at least one of the  
2        inserted procedures implements, when executed, a  
3        pre-defined function which is independent of  
4        functionality provided by the non-marked application  
5        program.

1       11. The apparatus in claim 6 wherein the security code  
2        provides functionality independent of any functionality  
3        provided by the application program.

1       12. The apparatus in claim 6 wherein the processor, in  
2        response to the stored instructions:  
3           (a) generates first and second separate flow  
4        representations for the first and second portions of the  
5        input executable code;  
6           (b) partitions each of the first and second flow  
7        representations into k-clusters each so as to yield first  
8        and second cluster flow representations, respectively  
9        (where k is a pre-defined integer);  
10          (c) randomly selects the first and second nodes in the  
11        first and second cluster flow representations,  
12        respectively, so as to form a corresponding one of the  
13        nodal pairs;  
14          (d) inserts a designation for the selected executable  
15        procedure at a first node in the nodal pair; and  
16          (e) repeats operations (c) and (d) a pre-defined number  
17        of times so as to insert a pre-defined number of separate  
18        procedures into the first and second flow representations  
19        so as to yield the combined flow representation.

1       13. The apparatus in claim 12 wherein the processor, in  
2       response to the stored instructions, inserts executable  
3       code for the selected one procedure in noncontiguous  
4       locations in the input executable code.

1       14. The apparatus in claim 12 wherein the processor, in  
2       response to the stored instructions, selects the one  
3       procedure from a pre-defined library of stored routines,  
4       wherein said one procedure is one of the stored routines.

1       15. The apparatus in claim 14 wherein each of the  
2       inserted procedures implements, when executed, a  
3       pre-defined function such that if any of said inserted  
4       procedures is removed from the marked code, the marked  
5       code, when subsequently executed, will terminate its  
6       execution.

1       16. The apparatus in claim 14 wherein at least one of  
2       the inserted procedures implements, when executed, a  
3       pre-defined function which is independent of  
4       functionality provided by the non-marked application  
5       program.

1       17. The apparatus in claim 12 wherein the processor, in  
2       response to the stored instructions:

3              randomly selects a node, U, in the first cluster  
4       flow representation;

5              randomly selects, with probability  $1-\lambda$  (where  $\lambda$  is a  
6       pre-defined value with  $0 \leq \lambda \leq 1$ ), a node Y in the second  
7       cluster flow representation;

DRAFT - 4/16/2020

8           randomly selects, with probability  $\lambda$ , a node Z,  
9   other than U, in the first cluster flow representation;  
10   and

11           provides designations of nodes Y and Z as the nodes  
12   forming the nodal pair.

1       18. The apparatus in claim 12 wherein the processor, in  
2   response to the stored instructions, randomly selects the  
3   first and second nodes from different clusters solely  
4   within the first cluster flow representation or from  
5   different clusters solely within the second cluster flow  
6   representation.

1       19. The apparatus in claim 4 wherein the processor, in  
2   response to the stored instructions:

3       (a) partitions the flow representation into k-clusters  
4   each so as to yield a cluster flow representation;  
5       (b) randomly selects the first and second nodes in the  
6   cluster flow representation so as to form a corresponding  
7   one of the nodal pairs;  
8       (c) inserts the designation for the selected executable  
9   procedure at a first node in the nodal pair;  
10      (d) repeats operations (b) and (c) a pre-defined number  
11   of times so as to insert a pre-defined number of separate  
12   procedures into the flow representation so as to yield  
13   the combined flow representation.

1       20. The apparatus in claim 19 wherein the processor, in  
2   response to the stored instructions, inserts executable  
3   code for the selected one procedure in noncontiguous  
4   locations in the input executable code.

1       21. The apparatus in claim 19 wherein the processor, in  
2       response to the stored instructions, selects the  
3       procedure from a pre-defined library of stored routines,  
4       wherein said procedure is one of the stored routines.

1       22. The apparatus in claim 21 wherein each of the  
2       inserted procedures implements, when executed, a  
3       pre-defined function such that if any of said inserted  
4       procedures is removed from the marked code, the marked  
5       code, when subsequently executed, will terminate its  
6       execution.

1       23. The apparatus in claim 21 wherein at least one of  
2       the inserted procedures implements, when executed, a  
3       pre-defined function which is independent of  
4       functionality provided by the non-marked application  
5       program.

1       24. The apparatus in claim 19 wherein the processor, in  
2       response to the stored instructions, randomly selects the  
3       first and second nodes from different clusters within the  
4       cluster flow representation.

1       25. For use with a computer system having a processor  
2       and a memory, the memory having computer executable  
3       instructions stored therein, a method for forming an  
4       identifier for input executable code and for securely  
5       marking the input executable code with the identifier so  
6       as to yield marked code, the method comprising the steps  
7       of:

8           generating a flow representation for the input  
9 executable code, the representation having a plurality of  
10 nodes, said nodes representing instructions in the input  
11 executable code, and connections among the nodes  
12 signifying associated control flow among instructions in  
13 the executable code;

14           randomly selecting first and second nodes from the  
15 plurality of nodes in the representation so as to form a  
16 pre-defined number of nodal pairs, each of said pairs  
17 having one of the first nodes and a corresponding one of  
18 the second nodes; and

19           for each of the nodal pairs, establishing execution  
20 flow between the first and second nodes in said each  
21 nodal pair and inserting, in the execution flow so  
22 established, executable code for a selected one of a  
23 plurality of different pre-defined executable procedures  
24 so as to collectively define the marked code, whereby the  
25 marked code contains the input executable code and a  
26 plurality of different ones of the pre-defined procedures  
27 each of which has been randomly spliced into control flow  
28 of the input executable code, wherein the identifier  
29 collectively comprises the executable code, for all the  
30 different ones of the plurality of predefined procedures,  
31 and the associated execution flows associated therewith  
32 and involving the nodal pairs.

1       26. The method in claim 25 wherein the input object  
2 comprises a software object.

1       27. The method in claim 26 wherein the software object  
2 comprises input executable code, at least one instruction

3       in the input executable code is associated with a  
4       corresponding one of the predefined first operation, and  
5       executable code for a corresponding executable procedure  
6       is associated with each selected one of the predefined  
7       second operations.

1       28. The method in claim 27 wherein the establishing and  
2       inserting step comprises the steps of:

3             3       inserting a pre-defined number of separate links and  
4       designations for the selected ones of the procedures into  
5       the flow representation so as to yield a combined flow  
6       representation; and

7             7       converting, in response to said input executable  
8       code and executable code for said selected ones of the  
9       procedures, said combined flow representation into output  
10      executable code, said output executable code being the  
11      marked code.

1       29. The method in claim 28 wherein the input executable  
2       code comprises first and second portions thereof and the  
3       flow representation comprises first and second separate  
4       flow representations for the first and second portions of  
5       the input executable code, respectively.

1       30. The method in claim 29 wherein:

2             2       the first portion of the input executable code  
3       comprises at least a pre-defined portion of a non-marked  
4       application program; and

5             5       the second portion of the input executable code  
6       comprises a remaining portion of the non-marked

CONFIDENTIAL

7 application program or pre-defined executable security  
8 code.

1 31. The method in claim 30 further comprising the step  
2 of selecting the procedure from a pre-defined library of  
3 stored routines, wherein said procedure is one of the  
4 stored routines.

1 32. The method in claim 31 further comprising the step  
2 of inserting executable code for the selected one  
3 procedure in noncontiguous locations in the input  
4 executable code.

1 33. The method in claim 31 wherein each of the inserted  
2 procedures implements, when executed, a pre-defined  
3 function such that if any of said inserted procedures is  
4 removed from the marked code, the marked code, when  
5 subsequently executed, will terminate its execution.

1 34. The method in claim 31 wherein at least one of the  
2 inserted procedures implements, when executed, a  
3 pre-defined function which is independent of  
4 functionality provided by the non-marked application  
5 program.

1 35. The method in claim 30 wherein the security code  
2 provides functionality independent of any functionality  
3 provided by the application program.

DRAFTED BY THE STAFF

1       36. The method in claim 30 further comprising the steps  
2       of:  
3       (a) generating first and second separate flow  
4       representations for the first and second portions of the  
5       input executable code;  
6       (b) partitioning each of the first and second flow  
7       representations into k-clusters each so as to yield first  
8       and second cluster flow representations, respectively  
9       (where k is a pre-defined integer);  
10      (c) randomly selecting the first and second nodes in the  
11      first and second cluster flow representations,  
12      respectively, so as to form a corresponding one of the  
13      nodal pairs;  
14      (d) inserting a designation for the selected executable  
15      procedure at a first node in the nodal pair; and  
16      (e) repeating operations (c) and (d) a pre-defined  
17      number of times so as to insert a pre-defined number of  
18      separate procedures into the first and second flow  
19      representations so as to yield the combined flow  
20      representation.

1       37. The method in claim 36 further comprising the step  
2       of inserting executable code for the selected one  
3       procedure in noncontiguous locations in the input  
4       executable code.

1       38. The method in claim 36 further comprising the step  
2       of selecting the procedure from a pre-defined library of  
3       stored routines, wherein said procedure is one of the  
4       stored routines.

1       39. The method in claim 38 wherein each of the inserted  
2       procedures implements, when executed, a pre-defined  
3       function such that if any of said inserted procedures is  
4       removed from the marked code, the marked code, when  
5       subsequently executed, will terminate its execution.

1       40. The method in claim 38 wherein at least one of the  
2       inserted procedures implements, when executed, a  
3       pre-defined function which is independent of  
4       functionality provided by the non-marked application  
5       program.

1       41. The method in claim 36 wherein the first and second  
2       nodes randomly selecting step comprises:

3             randomly selecting a node, U, in the first cluster  
4       flow representation;

5             randomly selecting, with probability  $1-\lambda$  (where  $\lambda$  is  
6       a pre-defined value with  $0 \leq \lambda \leq 1$ ), a node Y in the  
7       second cluster flow representation;

8             randomly selecting, with probability  $\lambda$ , a node Z,  
9       other than U, in the first cluster flow representation;  
10      and

11      providing designations of nodes Y and Z as the nodes  
12      forming the nodal pair.

1       42. The method in claim 36 wherein the first and second  
2       nodes randomly selecting step comprises the step of  
3       randomly selecting the first and second nodes from  
4       different clusters solely within the first cluster flow

DRAFT - PENDING EXAMINER'S REVIEW

5 representation or from different clusters solely within  
6 the second cluster flow representation.

1 43. The method in claim 28 further comprising the steps  
2 of:

3 (a) partitioning the flow representation into k-clusters  
4 each so as to yield a cluster flow representation;  
5 (b) randomly selecting the first and second nodes in the  
6 cluster flow representation so as to form a corresponding  
7 one of the nodal pairs;  
8 (c) inserting the designation for the selected  
9 executable procedure at a first node in the nodal pair;  
10 and  
11 (d) repeating operations (b) and (c) a pre-defined  
12 number of times so as to insert a pre-defined number of  
13 separate procedures into the flow representation so as to  
14 yield the combined flow representation.

1 44. The method in claim 43 further comprising the step  
2 of inserting executable code for the selected one  
3 procedure in noncontiguous locations in the input  
4 executable code.

1 45. The method in claim 43 further comprising the step  
2 of selecting the procedure from a pre-defined library of  
3 stored routines, wherein said procedure is one of the  
4 stored routines.

1 46. The method in claim 45 wherein each of the inserted  
2 procedures implements, when executed, a pre-defined  
3 function such that if any of said inserted procedures is

4       removed from the marked code, the marked code, when  
5       subsequently executed, will terminate its execution.

1       47. The method in claim 45 wherein at least one of the  
2       inserted procedures implements, when executed, a  
3       pre-defined function which is independent of  
4       functionality provided by the non-marked application  
5       program.

1       48. The method in claim 43 wherein the first and second  
2       nodes randomly selecting step comprises the step of  
3       randomly selecting the first and second nodes from  
4       different clusters within the cluster flow  
5       representation.

1       49. A computer readable medium having computer  
2       executable instructions stored therein for performing the  
3       steps of claim 25.

1       50. Executable computer code securely marked with an  
2       identifier and generated by a computer system, the system  
3       having a processor and a memory, the memory having  
4       computer executable instructions stored therein,  
5       characterized by the code having being produced by the  
6       steps, implemented by the processor in response to the  
7       executable instructions, recited in claim 25.